The Complexity of the Kth Largest Subset Problem and Related Problems

Christoph Haase[a,1], Stefan Kiefer[b,2]

[a] *LSV, CNRS & ENS Cachan, Université Paris-Saclay, France*
[b] *Department of Computer Science, University of Oxford, UK*

Abstract

We show that the KTH LARGEST SUBSET problem and the KTH LARGEST m-TUPLE problem are in PP and hard for PP under polynomial-time Turing reductions. Several problems from the literature were previously shown NP-hard via reductions from those two problems, and by our main result they become PP-hard as well. We also provide complementary PP-upper bounds for some of them.

Keywords: Kth largest subset, Kth largest m-tuple, counting problems, PP, computational complexity

1. Introduction

The following two problems are listed in Garey and Johnson's classical compendium on the theory of NP-completeness [1, p. 225] as problems [SP20] and [SP21], respectively:

KTH LARGEST SUBSET PROBLEM
INSTANCE: A set $X = \{x_1, \ldots, x_m\} \subseteq \mathbb{N}$ and $K, B \in \mathbb{N}$.
QUESTION: Is $\#\left\{Y \subseteq X : \sum_{x \in Y} x \leq B\right\} \geq K$?

KTH LARGEST m-TUPLE PROBLEM
INSTANCE: Finite sets $X_1, \ldots, X_m \subseteq \mathbb{N}$ and $K, B \in \mathbb{N}$.
QUESTION: Is $\#\left\{(x_1, \ldots, x_m) \in X_1 \times \cdots X_m : \sum_{i=1}^{m} x_i \geq B\right\} \geq K$?

[1]Supported by Labex Digicosme, Univ. Paris-Saclay, project VERICONISS.
[2]Supported by a University Research Fellowship of the Royal Society.

The first problem was initially introduced by Johnson and Kashdan [2] and the second by Johnson and Mizoguchi [3]. Garey and Johnson [1] state that both problems are NP-hard under polynomial-time Turing reductions but mention that their membership in NP, and, thus, their NP-completeness, is open. For the NP-hardness proof of the KTH LARGEST SUBSET problem, see Theorem 6 in the paper of Johnson and Kashdan [2], or the discussion provided by Garey and Johnson [1, p. 115]. It is unknown whether this problem is also NP-hard under polynomial-time many-one reductions [4, p. 148]. The KTH LARGEST SUBSET problem has its name for historical reasons: note that the the problem is about the Kth *smallest* subset.

Both problems have repeatedly been used as starting points for proving other problems NP-hard. In this note, we show that the KTH LARGEST SUBSET and the KTH LARGEST m-TUPLE problem are complete for the complexity class PP. More precisely, they are in PP, and they are hard for PP under polynomial-time Turing reductions. The complexity class PP [5] can be defined as the class of languages $L \subseteq \Sigma^*$ that have an NP Turing machine M_L such that for all words $w \in \Sigma^*$ one has $w \in L$ if and only if more than half of the computation paths of M_L on w are accepting. The class PP includes NP [5]. Closely related is the function class #P, which consists of those functions $f : \Sigma^* \to \mathbb{N}$ for which there exists an NP Turing machine M_f such that for all words $w \in \Sigma^*$ the function value $f(w)$ is equal to the number of accepting computation paths of M_f on w. The class PP is remarkably powerful: Toda's Theorem [6] states that P^{PP} (which equals $\mathrm{P}^{\#\mathrm{P}}$) contains the polynomial-time hierarchy PH. This means, in particular, that if the KTH LARGEST SUBSET (or m-TUPLE, resp.) problem were in NP then the polynomial-time hierarchy would collapse to P^{NP}. This can be taken as evidence (under standard assumptions from complexity theory) that the problems considered in this note are not in NP and, hence, are harder than NP.

By an application of the main result of this note, it follows that several other problems for which reductions from the KTH LARGEST SUBSET (or m-TUPLE) problem have been constructed are not only NP-hard under polynomial-time Turing reductions, but in fact PP-hard under polynomial-time Turing reductions. We give examples of such problems in Section 3.

2. Main Result

The class PP can equivalently be characterised as the class of languages $L \subseteq \Sigma^*$ that have a polynomial-time bounded deterministic Turing machine M with access to a fair coin such that for all $w \in \Sigma^*$ we have $w \in L$ if and only if M accepts w with probability at least $1/2$. This follows from Theorem 4.4 of Simon [7], and in particular from the fact that PP is closed under complement [7], see also the work of Gill [5]. However, in some proofs it will be more convenient if the Turing machine has access to *biased* rather than fair coins, and if the biases (rational numbers between 0 and 1) can depend on the input. More precisely, a *probabilistic polynomial-time bounded Turing machine (PP-TM)* is a polynomial-time bounded Turing machine that takes an input w and then operates in two phases:

1. Depending only on w, it deterministically computes finitely many rational numbers $b_1, \ldots, b_k$ with $b_i \in (0, 1)$ for all $i \in \{1, \ldots, k\}$. Each b_i is represented as a quotient of integers written in binary. We view the b_i values as *biases* of biased coins.
2. It acts like a nondeterministic polynomial-time bounded Turing machine, but the nondeterminism is resolved probabilistically using biased coins with biases $b_1, \ldots, b_k$ that were computed in the first phase. More concretely, besides deterministic instructions, the Turing machine has instructions of the form

   `if coin`(b_i) `then goto c1 else goto c2`

   where b_i is a bias computed in the first phase. The semantics of this operation is the natural one: with probability b_i the Turing machine continues at `c1`, otherwise (i.e., with probability $1 - b_i$) it continues at `c2`.

We have the following lemma.

Lemma 1. *Let $L \subseteq \Sigma^*$ and M be a PP-TM. Furthermore, let $\tau : \Sigma^* \to (0, 1)$ be a polynomial-time computable function returning a rational number encoded as a quotient of integers in binary such that for all $w \in \Sigma^*$ we have $w \in L$ if and only if M accepts w with probability at least $\tau(w)$. Then $L \in PP$.*

Proof. The proof consists of three steps:

1. We show that we can, without loss of generality, take $\tau(w) = 1/2$ for all $w \in \Sigma^*$.
2. We construct from M an NP Turing machine M' such that for all $w \in \Sigma^*$ we have $w \in L$ if and only if at least half of the computation paths of M' on w are accepting.
3. We argue that the existence of M' implies membership of L in PP.

In the first step of the proof, we show that we can, without loss of generality, take $\tau(w) = 1/2$ for all $w \in \Sigma^*$. Indeed, given M and τ, construct a PP-TM M' as follows. Given $w \in \Sigma^*$, the PP-TM M' first computes $\tau := \tau(w)$. If $\tau < 1/2$, it computes $p := (1/2 - \tau)/(1 - \tau)$, and accepts immediately with probability p. Otherwise (i.e., with probability $1 - p$) the PP-TM M' simulates M. It follows that

$$\Pr(M' \text{ accepts } w) = p + (1 - p) \cdot \Pr(M \text{ accepts } w).$$

A simple calculation shows that $\Pr(M \text{ accepts } w) \geq \tau(w)$ if and only if $\Pr(M' \text{ accepts } w) \geq 1/2$.

Similarly, if $\tau > 1/2$, the PP-TM M' computes $p := 1/(2\tau)$, and rejects immediately with probability $1 - p$. Otherwise (i.e., with probability p) the PP-TM M' simulates M. It follows that

$$\Pr(M' \text{ accepts } w) = p \cdot \Pr(M \text{ accepts } w).$$

Again, we have $\Pr(M \text{ accepts } w) \geq \tau(w)$ if and only if $\Pr(M' \text{ accepts } w) \geq 1/2$.

Consequently, for the remainder of the proof we can assume that for all $w \in \Sigma^*$ we have $w \in L$ if and only if M accepts w with probability at least $1/2$. Furthermore, we can ensure that for any $w \in \Sigma^*$ all computation paths of M on w involve the same number, say p_w, of accesses to a coin.

In the second step of the proof, we construct from M an NP Turing machine M' such that for all $w \in \Sigma^*$ we have $w \in L$ if and only if at least half of the computation paths of M' on w are accepting. Given a word $w \in \Sigma^*$, the Turing machine M' computes, like the PP-TM M, the set B of biases. Then M' computes a multiple, say q_w, of the denominators of the biases in B. Subsequently, M' simulates M, except that when it hits an instruction

$$\texttt{if coin}(b)\ \texttt{then goto c1 else goto c2}$$

then M' takes $b \cdot q_w$ nondeterministic branches to `c1` and $(1 - b) \cdot q_w$ nondeterministic branches to `c2`. By the definition of q_w the numbers $b \cdot q_w$

and $(1-b)\cdot q_w$ are natural numbers. They might be of exponential magnitude, but their representation in binary is of polynomial length. Using this representation, the Turing machine M' can implement the $b \cdot q_w$ and $(1-b)\cdot q_w$ nondeterministic branches using cascades of binary nondeterministic branches. By this construction, we have for any $w \in \Sigma^*$ that the total number of computation paths of M' on w equals $(q_w)^{p_w}$ and, furthermore, if M accepts w with probability x then $x \cdot (q_w)^{p_w}$ computation paths of M' on w are accepting. We conclude that we have $w \in L$ if and only if at least half of the computation paths of M' on w are accepting.

For the third and final step of the proof, construct an NP Turing machine M'' from M' by swapping accepting and rejecting states. Then we have $w \notin L$ if and only if *more than half* of the computation paths of M'' on w are accepting. Hence, using the definition of PP, the complement of L is in PP. But PP is closed under complement, see [5], thus $L \in \mathrm{PP}$. □

In order to show PP-hardness of the KTH LARGEST SUBSET and the KTH LARGEST m-TUPLE problem, we employ #P-completeness of a variant of the classical SUBSETSUM problem. Given a set $X = \{x_1, \ldots, x_m\} \subseteq \mathbb{N}$ and $T \in \mathbb{N}$, #SUBSETSUM asks for determining the number of subsets Y of X whose elements sum up to T, i.e., computing

$$\#\left\{ Y \subseteq X : \sum_{x \in Y} x = T \right\}.$$

Building upon the results of Hunt et al. [8], Faliszewski and Hemaspaandra [9, p. 104] derive that #SUBSETSUM is #P-complete, where #P-hardness holds under parsimonious many-one reductions. As defined by Simon [7], given functions $f, g : \Sigma^* \to \mathbb{N}$ we say that f *parsimoniously reduces to* g if there is a polynomial-time computable function $h : \Sigma^* \to \Sigma^*$ such that $f(w) = g(h(w))$ for all $w \in \Sigma^*$. For subsequent reference, let us capture the result of Faliszewski and Hemaspaandra09 [9] by the following proposition.

Proposition 2. #SUBSETSUM *is* #P*-complete, and in particular* #P*-hard under parsimonious many-one reductions.*

We are now prepared to prove our main result. In the remainder of this note, whenever we refer to PP-completeness, PP-hardness holds under polynomial-time Turing reductions. In the statements of the theorems we make this explicit to avoid a confusion of casual readers.

Theorem 3. *The* KTH LARGEST SUBSET *problem and the* KTH LARGEST m-TUPLE *problem are PP-complete under polynomial-time Turing reductions.*

PROOF. First, we prove membership in PP for both problems. Given an instance $X = \{x_1, \ldots, x_m\} \subseteq \mathbb{N}$ and $B, K \in \mathbb{N}$ of the KTH LARGEST SUBSET problem, we can construct a PP-TM M that first chooses every subset $Y \subseteq X$ uniformly at random by selecting every x_i with probability $1/2$. Subsequently, M accepts if and only if $\sum_{x \in Y} x \leq B$. It follows that the probability that M accepts is equal to

$$\# \left\{ Y \subseteq X : \sum_{x \in Y} x \leq B \right\} / 2^m.$$

Hence, $\#\{Y \subseteq X : \sum_{x \in Y} x \leq B\} \geq K$ if and only if M accepts with probability at least $K/2^m$. By Lemma 1, the KTH LARGEST SUBSET problem is in PP.

For the KTH LARGEST m-TUPLE problem, the construction is similar: a PP-TM M selects an m-tuple $(x_1, \ldots, x_m) \in X_1 \times \cdots \times X_m$ uniformly at random, i.e., each m-tuple with probability $1 \big/ \prod_{i=1}^{m} (\# X_i)$. By the characterisation of PP in Lemma 1, this can be achieved by independently choosing some $x_i \in X_i$ with probability $1/\# X_i$ for every $1 \leq i \leq m$. Then, M accepts if and only if $\sum_{i=1}^{m} x_i \geq B$. Hence, $\# \{(x_1, \ldots, x_m) \in X_1 \times \cdots \times X_m : \sum_{i=1}^{m} x_i \geq B\} \geq K$ if and only if M accepts with probability at least $K \big/ \prod_{i=1}^{m} (\# X_i)$. An application of Lemma 1 shows that the KTH LARGEST m-TUPLE problem is in PP as well.

In order to show hardness for PP, we give polynomial-time Turing reductions from the canonical PP-complete problem MAJSAT [5, 10]. This problem asks, given a Boolean formula ψ over n Boolean variables $b_1, \ldots, b_n$, does a majority (i.e., at least $2^{n-1} + 1$) of the variable assignments satisfy ψ. Since #SAT, i.e., the task of computing the number of satisfying assignments of a Boolean formula, is #P-complete [10], it follows from Proposition 2 that there is a polynomial-time computable function h such that $h(\psi)$ outputs an instance $X = \{x_1, \ldots, x_m\}$ and $T \in \mathbb{N}$ of #SUBSETSUM such that

$$\begin{aligned} L &:= \# \left\{ Y \subseteq X : \sum_{x \in Y} x = T \right\} \\ &= \# \left\{ (b_1, \ldots, b_n) \in \{0,1\}^n : \psi(b_1, \ldots, b_n) = 1 \right\}. \end{aligned}$$

Using binary search, with at most $2 \cdot m$ queries to the KTH LARGEST SUBSET problem we can compute

$$M := \# \left\{ Y \subseteq X : \sum_{x \in Y} x \leq T \right\} \quad \text{and} \quad N := \# \left\{ Y \subseteq X : \sum_{x \in Y} x \leq T - 1 \right\}.$$

We now have that $L = M - N$, and hence a majority of the variable assignments satisfies ψ if and only if $M - N \geq 2^{n-1} + 1$.

In the same fashion, we can show hardness of the KTH LARGEST m-TUPLE problem. Set $X_i := \{0, x_i\}$ for every $1 \leq i \leq m$. As before, using binary search with at most $2 \cdot m$ queries we can compute

$$\begin{aligned} M &:= \# \left\{ (x_1, \ldots, x_m) \in X_1 \times \cdots \times X_m : \sum_{i=1}^{m} x_i \geq T \right\} \\ &= \# \left\{ Y \subseteq X : \sum_{x \in Y} x \geq T \right\} \end{aligned}$$

and

$$\begin{aligned} N &:= \# \left\{ (x_1, \ldots, x_m) \in X_1 \times \cdots \times X_m : \sum_{i=1}^{m} x_i \geq T + 1 \right\} \\ &= \# \left\{ Y \subseteq X : \sum_{x \in Y} x \geq T + 1 \right\}. \end{aligned}$$

As above, $L = M - N$ and a majority of the variable assignments satisfies ψ if and only if $M - N \geq 2^{n-1} + 1$, from which PP-hardness of the problem follows. □

3. Applications

In this section, we discuss several problems whose NP-hardness has been shown in the literature by reductions from the KTH LARGEST SUBSET (or m-TUPLE) problem. By Theorem 3, all of these problems become PP-hard. As mentioned in Section 1, Toda's Theorem implies that if any of these problems were in NP (and hence NP-complete), the polynomial-time hierarchy would collapse to P^{NP}.

3.1. The KTH LARGEST-AREA CONVEX POLYGON *Problem*

Chazelle [11] posed the following question: "[G]iven n points in the Euclidean plane, how hard is it to compute the Kth largest-area convex polygon formed by any subset of the points?" This question was essentially answered by Salowe [12]. In particular, the following decision variant was considered in [12]:

KTH LARGEST-AREA CONVEX POLYGON
INSTANCE: A set P of n points $(x_1, x_2) \in \mathbb{N} \times \mathbb{N}$, and $K, B \in \mathbb{N}$.
QUESTION: Are there at least K subsets $P' \subseteq P$ for which $P' = extr(conv(P'))$ and the area of $conv(P')$ is at least B ?

Here, $conv(P')$ refers to the convex hull of the points in P', and $extr(conv(P'))$ refers to the set of extreme points in that convex hull. Salowe [12] showed that the KTH LARGEST-AREA CONVEX POLYGON problem is NP-hard, via a reduction from the KTH LARGEST m-TUPLE problem. In fact, we have:

Theorem 4. *The* KTH LARGEST-AREA CONVEX POLYGON *problem is PP-complete under polynomial-time Turing reductions.*

PROOF. PP-hardness follows from the reduction in [12] combined with Theorem 3. Towards membership in PP, construct a PP-TM M, which, given an instance of the KTH LARGEST-AREA CONVEX POLYGON problem, selects a subset P' of P uniformly at random, and then checks (in polynomial time) whether $P' = extr(conv(P'))$ and the area of $conv(P')$ is at least B. If yes, M accepts, otherwise M rejects. So the acceptance probability is at least $K/2^n$ if and only if the given instance of the KTH LARGEST-AREA CONVEX POLYGON problem is a yes-instance. An application of Lemma 1 shows membership in PP. □

Salowe [12] remarked that the corresponding enumeration problem is #P-complete. From this observation it also follows that the KTH LARGEST-AREA CONVEX POLYGON problem is PP-hard, although Salowe [12] only claims NP-hardness. He also says "it is not clear that [the KTH LARGEST-AREA CONVEX POLYGON problem] is in NP". But this seems unlikely in view of Theorem 4.

3.2. The VERTEX COUNTING *Problem*

Let $m, n \in \mathbb{N}$ with $m > n$. Let $b \in \mathbb{Q}^m$ be a column vector and $A \in \mathbb{Q}^{m \times n}$ be a rational $m \times n$ matrix with its rows denoted by $a_1, \ldots, a_m \in \mathbb{Q}^n$. Then A and b define a polyhedron $P := \{x \in \mathbb{R}^n : A \cdot x \leq b\}$. A *vertex* of P is a point that is the unique intersection of at least n of the bounding hyperplanes $a_i \cdot x = b_i$, $i \in \{1, \ldots, m\}$. The following problem was considered by Dyer [13]:

VERTEX COUNTING

INSTANCE: A matrix $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^n$, and $K \in \mathbb{N}$.

QUESTION: Does the polyhedron $A \cdot x \leq b$ have at most K vertices ?

Dyer [13, Proposition 3] has shown that the VERTEX COUNTING problem is NP-hard by a Turing reduction from the KTH LARGEST SUBSET problem. In fact, we have:

Theorem 5. *The* VERTEX COUNTING *problem is PP-complete under polynomial-time Turing reductions.*

PROOF. PP-hardness follows from the reduction in [13] combined with Theorem 3. To prove membership in PP, we use the fact that PP is closed under complement [5]. So it suffices to prove PP-membership of the CO-VERTEX COUNTING problem, which is like the VERTEX COUNTING problem except that it asks whether the polyhedron has *at least* K vertices. Construct a PP-TM M, which, given an instance of the CO-VERTEX COUNTING problem, selects n bounding hyperplanes, say $h_1, \ldots, h_n$, uniformly at random, and then checks whether they have a unique intersection. If no, M rejects. Otherwise M computes the unique intersection, say p, and checks whether p is in the polyhedron. If no, M rejects. Otherwise M computes the set S of all bounding hyperplanes on which p lies. Then M computes the lexicographically smallest n-element subset S' of S such that p is the unique intersection of the hyperplanes in S'. That set S' can be computed in polynomial time, e.g., by greedily adding hyperplanes with linearly independent normal vectors. If $S' = \{h_1, \ldots, h_n\}$, then M accepts, otherwise M rejects. By construction, any vertex p is accepted by exactly one run of M. So the acceptance probability is at least $K \big/ \binom{m}{n}$ if and only if the given instance of the CO-VERTEX COUNTING problem is a yes-instance. An application of Lemma 1 shows membership in PP. □

3.3. Routing Flows with Delay Guarantees

In [14] the problem of routing flows through networks is considered, where the information available for making routing decisions is probabilistic. The aim of the routing algorithm is to guarantee certain quality-of-service requirements. One of the specific problems is to evaluate the *end-to-end delay* of a given path, as we describe in the following.

Let $e_1, \ldots, e_n$ be a path in a network. For each $i \in \{1, \ldots, n\}$, the number $d_i \in \mathbb{N}_0$ is a random *delay* introduced by the link e_i. For all $i \in \{1, \ldots, n\}$ and $k \in \mathbb{N}_0$, the probability that $d_i = k$ is given by $p_{i,k} \in [0,1] \cap \mathbb{Q}$. It is assumed that for each i there are only finitely many k with $p_{i,k} > 0$, and that the delays d_i are independent. The *end-to-end delay* is $\sum_{i=1}^{n} d_i$. An *end-to-end delay requirement* is given by a maximum delay $D \in \mathbb{N}_0$ and a probability threshold $\tau \in [0,1] \cap \mathbb{Q}$. Guerin and Orda refer to the following problem as "problem $P(\pi)$" [14]:

DELAY GUARANTEE

INSTANCE: A path length $n \in \mathbb{N}$, a list of all positive $p_{i,k} \in [0,1] \cap \mathbb{Q}$, the maximum delay $D \in \mathbb{N}_0$, a probability threshold $\tau \in [0,1] \cap \mathbb{Q}$.

QUESTION: Is the probability of $\sum_{i=1}^{n} d_i \leq D$ at least τ ?

In [14, Lemma IV.1] the DELAY GUARANTEE problem was shown NP-hard by a reduction from the KTH LARGEST SUBSET problem. In fact, we have:

Theorem 6. *The* DELAY GUARANTEE *problem is PP-complete under polynomial-time Turing reductions.*

PROOF. PP-hardness follows from the reduction in [14] combined with Theorem 3. Towards membership in PP, construct a PP-TM M, which, given an instance of the DELAY GUARANTEE problem, takes independent random values for the d_i according to the given $p_{i,k}$ and then checks whether $\sum_{i=1}^{n} d_i \leq D$. If yes, M accepts, otherwise M rejects. So the acceptance probability is at least τ if and only if the given instance of the DELAY GUARANTEE problem is a yes-instance. An application of Lemma 1 shows membership in PP. □

3.4. Further Probabilistic Problems

The KTH LARGEST SUBSET problem has been used by Laroussinie and Sproston [15] to show NP-hardness and coNP-hardness of model-checking "durational probabilistic systems" (a variant of timed automata) against the

probabilistic and timed temporal logic PTCTL. Combining this reduction (which is similar to the one for the DELAY GUARANTEE problem) with Theorem 3 strengthens the NP-hardness and coNP-hardness results of [15] to PP-hardness.

Finally, the KTH LARGEST m-TUPLE problem was used by Cire et al. [16] to show NP-hardness of another probabilistic problem, namely determining whether a so-called "chance-alldifferent constraint" has a solution. This is a weighted extension of stochastic constraint programming. Cire et al. [16] remarked: "Also, we are not aware if the problem of deciding whether there exists a feasible solution to chance-alldifferent is in NP." But this seems unlikely, as their reduction combined with Theorem 3 proves PP-hardness of this problem.

References

[1] M. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman & Co., New York, NY, USA, 1979.

[2] D. Johnson, S. Kashdan, Lower Bounds for Selection in $X+Y$ and Other Multisets, J. ACM 25 (4) (1978) 556–570.

[3] D. Johnson, T. Mizoguchi, Selecting the Kth Element in $X + Y$ and $X_1 + X_2 + \cdots + X_m$, SIAM J. Comput. 7 (2) (1978) 147–153.

[4] S. Homer, A. Selman, Computability and Complexity Theory, Second Edition, Texts in Computer Science, Springer, 2011.

[5] J. Gill, Computational Complexity of Probabilistic Turing Machines, SIAM J. Comput. 6 (4) (1977) 675–695.

[6] S. Toda, PP is as Hard as the Polynomial-Time Hierarchy, SIAM J. Comput. 20 (5) (1991) 865–877.

[7] J. Simon, On some central problems in computational complexity, Tech. Rep. 75-224, Cornell University, URL `http://hdl.handle.net/1813/6975`, 1975.

[8] H. Hunt, M. Marathe, V. Radhakrishnan, R. Stearns, The Complexity of Planar Counting Problems, SIAM J. Comput. 27 (4) (1998) 1142–1167.

[9] P. Faliszewski, L. Hemaspaandra, The complexity of power-index comparison, Theor. Comput. Sci. 410 (1) (2009) 101–107.

[10] C. H. Papadimitriou, Computational Complexity, Addison-Wesley, 1994.

[11] B. Chazelle, New Techniques for Computing Order Statistics in Euclidean Space (Extended Abstract), in: Proc. Symp. Comput. Geom., ACM, 125–134, 1985.

[12] J. Salowe, Selecting the Kth largest-area convex polygon, in: Proc. WADS, vol. 382 of *LNCS*, Springer, 243–250, 1989.

[13] M. Dyer, The Complexity of Vertex Enumeration Methods, Math. Oper. Res. 8 (3) (1983) 381–402.

[14] R. Guérin, A. Orda, QoS Routing in Networks with Inaccurate Information: Theory and Algorithms, IEEE/ACM Trans. Netw. 7 (3) (1999) 350–364.

[15] F. Laroussinie, J. Sproston, Model Checking Durational Probabilistic Systems, in: Proc. FoSSaCS, vol. 3441 of *LNCS*, Springer, 140–154, 2005.

[16] A. Cire, E. Coban, W.-J. van Hoeve, Flow-Based Combinatorial Chance Constraints, in: Proc. CPAIOR, vol. 7298 of *LNCS*, Springer, 129–145, 2012.